Homework 8: Q3

Name: Waiwai Kim, waiwaiki

1 Part (a): Sub-part 1 Proof Idea

Here, we are asked to prove that the algorithm given to the problem takes $\Omega(n^2)$ to make the MST. The proof idea is given in the recitation notes. The idea is that if the algorithm reads the entire input of n^2, the lower bound of the algorithm is n^2 regardless of what happens after reading the input. Said differently the way the algorithm reads the input is the limiting factor.

2 Part (a): Sub-part 2

Here, we are asked to show that it is incorrect to argue for the lower bound of an algorithm based on its need to read the entire input. The counter example is Binary Search Tree. The input size to BST is n; however, the run-time of BST is $O(logn)$. Thus, this counter example sufficiently shows that the argument in part (a) which is based on the size is not correct. We cannot say that the run time for finding MST is $\Omega(n^2)$ because there exists an input that cause the algorithm to run in less than n^2. For example, if a graph has n nodes in which node a_i has edges to a_{i-1} and a_{i+1}. In other words, this graph almost looks like a linear

3 Part (b): Proof of Idea

The idea here is for an adversary to fool Prim's algorithm. We are going to assume that the adversary knows Prims's algorithm. The adversary aims to provide an input X for the algorithm that maximizes the number of comparisons done by the algorithm. Note that Prim algorithm maintains a list called key that has the minimum distance edge across the cut between S and $V-S$. Note that the notations S is commonly used for the growing MST, while $V-S$ is used for the set excluding S. For a given node that was just added, the algorithm has to update key because the recently added node has edges across the cut. The idea of the adversarial argument is that we can carefully choose so that it maximizes the number of updating key for a given added node.

4 Part (b): Proof Details

Assume that the input has n nodes $n_1,...,n_n$. The initial number of unfixed distances of the edges is n^2. We will define the invariants in terms of the ith added node to MST. Let's call this n_i. Without loss of generality, we can assume or state that $i-1$ number of nodes were added before n_i. Let's call these nodes $n_1,...n_{i-1}$. Let's say key up to this point is key_{i-1}. Said differently, key updated after node n_i is added to MST is key_i. We will maintain the following invariants:

- the edge distances from nodes $n_1, n_2, ...n_{i-1}$ that have been added to MST are fixed.
- the edge distances from nodes that have not been added to MST are not fixed yet.

At adding node n_i, the adversary can choose the distances from n_i to $n_{i+1}, n_{i+2}, ..., n_n$ so that the minimum distances of the edges crossing the cut have to be updated, which is maintained by key. Note that the distance of an edge (n_i, n_k)

where $i \le k < n$ is already determined when a previous node n_k was added to MST. This is also stated by the first invariant above. Thus, at node n_i, $n - i + 1$ number of edges are determined and updated in key. Said differently, over all n nodes, the number of updates is $\Sigma_{i=1}^{n} n - i + 1$, which is $O(n^2)$.

Assume there exists an algorithm A which runs in $\Sigma_{i=1}^{n}(n - i + 1) - 1$ which correctly finds MST in an undirected graph G of size n. Let X be a graph where $d(n_i, n_k) = d(n_{i-1}, n_l) + 1$ where $i \le k < n$ and $i - 1 \le l < n$ so the key for k where $i \le k < n$ has to be updated. Run algorithm A on X. Since it takes $\Sigma_{i=1}^{n}(n - i + 1) - 1$ comparisons, there is at least one node of which the distance to its neighbor is not determined. For the ease of illustration, after $\Sigma_{i=1}^{n}(n - i + 1) - 1$ comparisons, we can say that the edge distance between node n_1 and n_n has not been determined without loss of generality. We will set $d(n_1, n_n)$ smaller than $d(n_{n-1}, n_n)$. Algorithm A returns MST in which node n_n is connected through n_{n-1}. However, the adversary can choose $d(n_1, n_n)$ so that the closet way to arrive at n_n is through n_1. Thus, the algorithm A is wrong. Thus, there cannot exist any correct MST algorithm that finds MST in an undirected graph G in less than $O(n^2)$ time.